FDREnet: Face Detection and Recognition Pipeline

Deepali Virmani
Department of Computer Science & Engineering, BPIT,
Guru Gobind Singh Indraprastha University, Delhi, India
hodcse@bpitindia.com

Palak Girdhar
Department of Computer Science & Engineering, BPIT,
Guru Gobind Singh Indraprastha University, Delhi, India
palakgirdhar@bpitindia.com

Prateek Jain
Department of Computer Science & Engineering, BPIT,
Guru Gobind Singh Indraprastha University, Delhi, India
prateek.jain2903@gmail.com

Pakhi Bamdev
Department of Computer Science & Engineering, BPIT,
Guru Gobind Singh Indraprastha University, Delhi, India
bpakhi97@gmail.com

***Abstract*—Face detection and recognition are being studied extensively for their vast applications in security, biometrics, healthcare, and marketing. As a step towards presenting an almost accurate solution to the problem in hand, this paper proposes a face detection and face recognition pipeline - face detection and recognition embedNet (FDREnet). The proposed FDREnet involves face detection through histogram of oriented gradients and uses Siamese technique and contrastive loss to train a deep learning architecture (EmbedNet). The approach allows the EmbedNet to learn how to distinguish facial features apart from recognizing them. This flexibility in learning due to contrastive loss accounts for better accuracy than using traditional deep learning losses. The dataset's embeddings produced from the trained FDREnet result accuracy of 98.03%, 99.57% and 99.39% for face94, face95, and face96 datasets respectively through SVM clustering. Accuracy of 97.83%, 99.57%, and 99.39% was observed for face94, face95, and face96 datasets respectively through KNN clustering.**

Keywords-convolution neural network; contrastive loss; histogram of oriented gradients; KNN clustering; Siamese technique; SVMcClustering

I. INTRODUCTION

The recognition of a face in an image is a tedious task as it depends on many factors like camera resolution, posture, face size, image brightness, lightening conditions, etc. Prior to face recognition, face detection is a necessity which is subjected to deep learning or classification for training and testing purposes. There are various techniques for face detection like skin color isolation [1, 2], Haar cascading [3], histogram of oriented gradients [4, 5]. After having the faces detected, the next step is to recognize them in any given frame or a picture. Various deep learning models [6-8] are provided to extract features of an image and indirectly teach the network how a particular face looks like. This approach was challenged first by FaceNet [9], which introduced a new type of loss called triplet loss that allowed deep learning models to find similarities and dissimilarities within different faces. FaceNet requires three inputs, an anchor (image in question), a positive (image of the same class as that of the anchor) and negative (image not from the same class as that of the anchor). Later a technique based on likeness loss called Siamese network was introduced which worked on two image inputs and a binary number defining if two images belong to the same class or not. FaceNet achieved state of art accuracy of 99.63% on LFW dataset while it achieved 95.12% of accuracy on YouTube Faces DB dataset. Face recognition based on convolution Siamese networks [10] used the Siamese network technique allowing the network to learn similarities and dissimilarities between faces. It achieved an accuracy of 98.63% on LFW dataset. Face detection and recognition using Viola-Jones with PCA-LDA and square Euclidean distance [11] worked on the face94 face95 and face96 datasets which will be also used in this paper. The approach discussed in this paper is face detection and recognition pipeline (FDREnet) which involves studying existing face detection methodologies, finding the best one for the task, and employ Siamese training technique to train our proposed deep learning network. After training embeddings of the dataset are produced from the trained model and SVM and KNN clustering is applied to achieve the best results.

II. LITERATURE SURVEY

Face detection is the most important step prior to face recognition. This section focuses on the comparative study of popular algorithms: skin color isolation [1, 2], Haar cascading [3], and histogram of oriented gradients [4, 5] to detect faces in an image. For this task, the open source datasets face94, face95, and face96 are used. Some random images from the datasets are shown in Figure 1. In the end, a brief approach to present face recognition algorithms is discussed.

Fig. 1. Sample images from face94, face95 and face96 datasets

Corresponding author: Prateek Jain

A. Skin Color Isolation

Detection of skin color in an image is useful in many areas like face and gesture recognition. Skin detection by segmentation is highly affected by various factors like illumination, artificial lighting vs sunlight, shadows, and camera resolution, therefore, skin pixel classification is not an easy task. The algorithm in [1, 2] involves selecting a range which will differentiate between skin and non-skin pixels. The range is decided such that most of the skin color values fall in that range so that different color skins can be segmented. HSV is an alternate color representation of the RGB color space. The algorithm to convert RGB color space to HSV color space is shown in Figure 2. HSV is designed in a way similar to human color perception. Therefore, HSV color space is a better approach to segment skin color than RGB color space. HSV is a cylindrical geometry which is defined by three components:

- Hue: Hue represents color and it ranges between 0 and 360
- Saturation: Saturation is the amount of grey in the color ranging between 0 (primary color) and 1 (gray). As we go towards the vertical axis of the cylindrical representation, pure colors start to fade.
- Value or brightness: It describes the brightness/intensity of the color and works in conjunction with saturation. It ranges between 0(dark) and 1.
- The steps of skin color isolation algorithm are shown in Figure 3.

The steps of skin color isolation algorithm are shown in Figure 3.

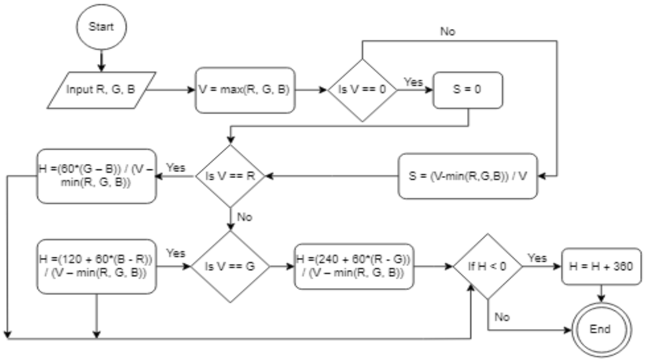


Fig. 2. RGB to HSV conversion flowchart

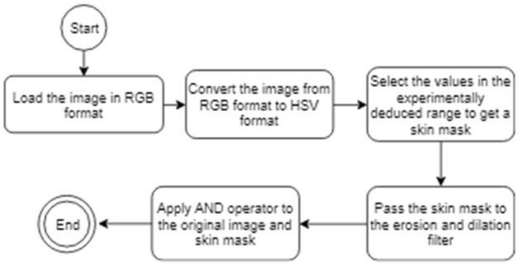

Start
Load the image in RGB format
Convert the image from RGB format to HSV format
Select the values in the experimentally deduced range to get a skin mask
Pass the skin mask to the erosion and dilation filter
Apply AND operator to the original image and skin mask
End

Fig. 3. Algorithm for face detection using skin color isolation

As we move from the top to bottom vertically, the color starts to be less bright. Pure colors are arranged at the outside edge of the cylindrical representation. After conducting a few experiments, the best values for this algorithm to work were in the range: 100-179 for Hue, 0-15 for Saturation and 65-255 for brightness. Some results of the skin color isolation algorithm are shown in Figure 4.

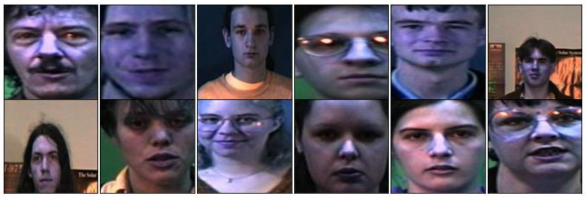


Fig. 4. Results of face detection using skin color isolation

B. Haar Cascading

Haar cascading is an object detection algorithm which can detect objects in images at high rates. It is a machine learning based approach which requires the data to train the model. Training data consist of positive images (images which contain objects to be detected) and negative images (images which don't contain those objects). After the classifier is trained, we can then use it to detect an object in other images. Haar cascading algorithm works in three steps [3]:

- Integral image: The algorithm uses features of the image to classify data. Features contain more information than individual image pixels. Moreover, a feature based model works faster than a pixel-based model. For features extraction from a given image, Haar features are used. Haar features are values obtained by subtracting the sum of pixels within the first rectangle from the sum of pixels within another rectangle. Possible sizes and locations of rectangles need to be considered, so, the number of features for an image is quite high.
- Constructing a classifier by selecting a small number of important features using AdaBoost: Most of the features calculated in the above step do not provide any useful information and are superfluous. Therefore, we need a method to dump those features which do not contain any relevant information. This is done using AdaBoost classifier which excludes a large number of available features and retains only the critical ones. The Haar cascading algorithm is shown in Figure 5.
- The attention cascade: There are many regions in the image which do not contain the object we want to detect and a lot of computational power is wasted on these regions. For this reason, the last step is to use a cascade of classifiers. Instead of applying all the selected features on the sub-region/window, the features are grouped into different stages of the classifier and are applied one by one. If a window fails the first stage, it is discarded. If the stage passes, second stage is applied and so on. If a window passes all the stages that means the window consists of the object. Figure 6 shows the results of Haar cascading algorithm.

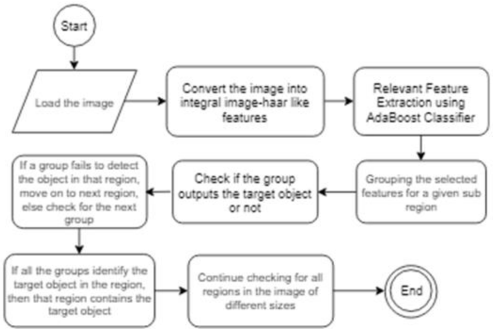

Start
Load the image
Convert the image into integral image-haar like features
Relevant Feature Extraction using AdaBoost Classifier
Grouping the selected features for a given sub region
Check if the group outputs the target object or not
If a group fails to detect the object in that region, move on to next region, else check for the next group
If all the groups identify the target object in the region, then that region contains the target object
Continue checking for all regions in the image of different sizes
End

Fig. 5. Algorithm for face detection using Haar cascading

Fig. 6. Results of face detection using Haar cascading

C. Histograms of Oriented Gradient (HOG)

HOG is an object detection algorithm which works by converting pixels into gradients [4, 5]. Gradients are arrows whose direction represents the change in pixel illumination from light to dark. To calculate the gradient of each pixel, the current pixel is compared to its neighboring pixels and the direction of change from light pixels to dark pixels is noted. The gradients are helpful in detecting target objects in conditions where there is a change in brightness. In face detection, a person's face will have different pixel values in light and dark conditions. But the change from light pixels to dark pixels remains almost the same. Therefore, gradients solve the problem of object detection even in dark conditions. It would be costly to calculate the gradient for each pixel. Therefore, a window is dragged on the image. Each window calculates the gradients and then the strongest gradient is chosen for that window. This process is continued for every window in the image. To detect an object in the image, the part of the test image which best matches up with the HOG representation of the trained image, contains the target object. The described algorithm is shown in Figure 7 and the results are shown in Figure 8.

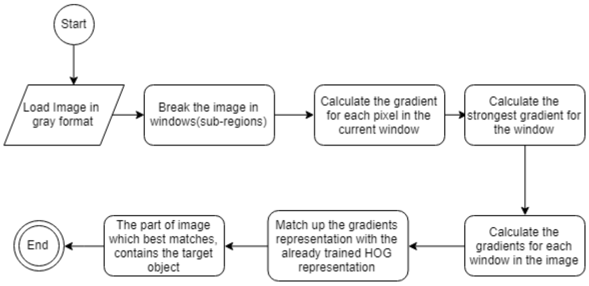

Start
Load Image in gray format
Break the image in windows(sub-regions)
Calculate the gradient for each pixel in the current window
Calculate the strongest gradient for the window
Calculate the gradients for each window in the image
Match up the gradients representation with the already trained HOG representation
The part of image which best matches, contains the target object
End

Fig. 7. Algorithm for face cetection using HOG

Fig. 8. Results of face detection using HOG

D. Comparative Study of Face Detection Methods

After using each of the above-mentioned face detection methods we conclude that HOG is the best algorithm for facial detection, especially when live video streaming is concerned. Though the above depiction of results of each algorithm looks alike, if one looks closer, the exactness of facial area covered by the detected frame is easily observed. Skin color detection has the flaw of taking the area that matches the skin tone. Using HSV tuning to detect skin tone colors, this algorithm can detect the similar colored objects as humans. Haar cascade and HOG have comparable results. For its ability to get zoomed in face features at a detected face, we preferred the HOG algorithm for the face detection task.

E. Face Recognition Methods

The most popular technique used for face recognition is deep learning. It was Google's facenet deep learning model [9] that introduced the new technique of training deep learning models via triplet input and triplet loss. This technique allowed the network to learn the features that distinguish two images instead of just categorizing alike images together. This technique produced a variant of Siamese network [10] that achieved the same task as that of triplet loss network by using just two input images and a binary value to define the relationship of the input images, whether they belong to the same class of not. The proposed approach is a Siamese technique used to train a proposed convolution neural network to achieve state of the art accuracy.

III. FDRENET

A. Framework

EmbedNet is the proposed approach for an efficient detection through HOG algorithm and a deep learning model aided by Siamese technique to differentiate and recognize the faces. The self-modeled deep learning convolution neural network (EmbedNet) is responsible for extracting features from the input images and is majorly responsible for producing embeddings for the recognition task. Here, the motive is to teach the network to differentiate between facial identities of different individuals as it simultaneously learns to find similarities between one individual's facial features, snapped at a different time with varying emotions. Doing so, the model not only learns to identify the images belonging to the same class but also tends to assimilate the features that mark one class's element relatively closer or farther from the other classes in terms of Euclidean distance of their encodings. Encodings may also be referred to as embeddings throughout this paper. To implement this idea, Siamese network with contrastive loss is favored. FDREnet after the detection task through HOG basically subsumes a pair of proposed identical convolution neural networks which is given the name EmbedNet. These twin networks which share the weights are trained by the Siamese technique. For the recognition task, the first sub part is to build a Siamese network architecture keeping the EmbedNet as the core deep learning model being trained. Siamese network takes in two input images which can be either a positive-positive pair or a positive-negative pair. Each of these EmbedNets receives one of the input images to produce an 128x1 encoding for both images. These encodings are then fed to the contrastive loss function, which governs the training of the network. Figure 9 provides a brief summary of the whole approach undertaken for the proposed FDREnet.

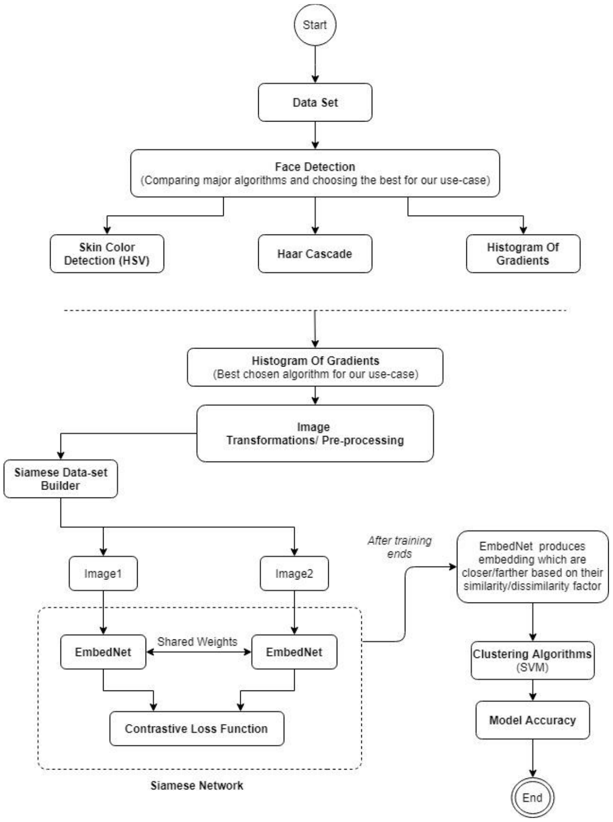

Start
Data Set
Face Detection
(Comparing major algorithms and choosing the best for our use-case)
Skin Color Detection (HSV)
Haar Cascade
Histogram Of Gradients
Histogram Of Gradients
(Best chosen algorithm for our use-case)
Image Transformations/ Pre-processing
Siamese Data-set Builder
Image1
Image2
EmbedNet
Shared Weights
EmbedNet
Contrastive Loss Function
Siamese Network
After training ends
EmbedNet produces embedding which are closer/farther based on their similarity/dissimilarity factor
Clustering Algorithms (SVM)
Model Accuracy
End

Fig. 9. The framework of the proposed EmbedNet

B. Loss Function for FDREnet

The objective of the Siamese network is to learn how to differentiate between two input images rather than performing a simple classification task. Thus, to train the network (see Figure 10), we can't use any loss function like cross entropy loss, which is primarily a classification loss function. The required loss function must have the ability to calculate the variance, which will account for learning the differences between the two images. The best fit for this use case scenario is contrastive loss. Intuitively, it evaluates how different two given images are as learned by the network. The embeddings of the images obtained from the EmbedNet are sent to Euclidian distance function (1) which finds the distance between the two embeddings.

$$d(p,n) = \sqrt{(n-p)^2 + (n+p)^2} \qquad (1)$$

where p is the first image's encoding, n is the second image's encoding and $d(p,n)$ is the calculated Euclidean distance.

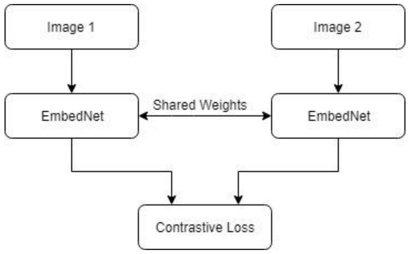


Fig. 10. Siamese network architecture

$$Loss = (1-y)(0.5)\big(d(n,p)\big)^2 -$$
$$(y)(0.5)\{0, \max(0, margin - d(n,p))\}^2 \qquad (2)$$

Equation (2) displays the contrastive loss function. Here y is 1 if the two images belong to the different class, it is 0. The margin is a value that we decided to it keep as 2. It is a value greater than 0 that helps to determine when not to let contrastive loss contribute to the total loss. Margin indicates that any value beyond this positive integer must not add up to the total loss. This optimizes network to consider image pairs that seem to be alike but are actually dissimilar. Figure 11 summarizes the use of contrastive loss and assignment of value to y.

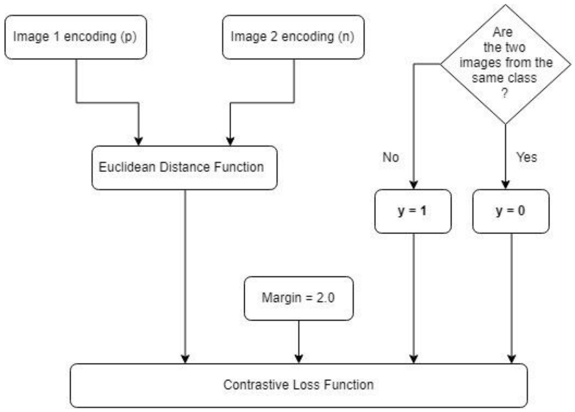


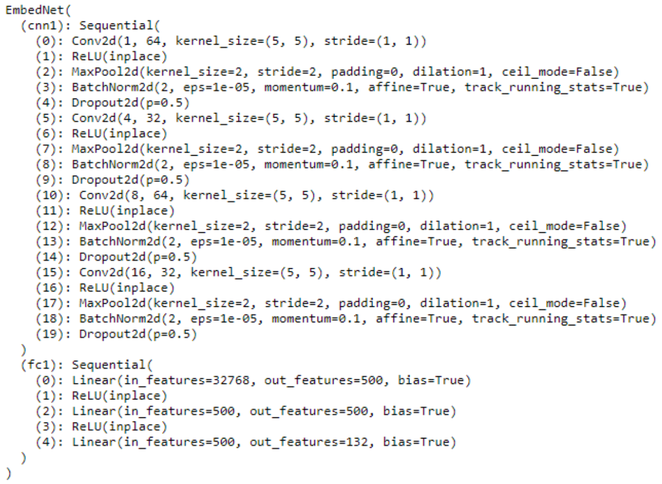
Image 1 encoding (p)
Image 2 encoding (n)
Are the two images from the same class ?
No
Yes
Euclidean Distance Function
y = 1
y = 0
Margin = 2.0
Contrastive Loss Function

Fig. 11. Inputs to contrastive loss function

This loss helps to train the inner embedding model (EmbedNet) in such a way that facial identities (images) belonging to the same individual will have closer embeddings and will be farther from the embeddings of other individual's facial identities.

C. FDREnet's Convolution Neural Network: EmbedNet

As discussed above, FDREnet subsumes two identical EmbedNets having shared weights. EmbedNet is a simple convolution neural network which is solely responsible for extracting features from the input images. The extracted features are the latent features of the input image as produced by the last output layer of the EmbedNet by a feed forward operation. We used four convolution layers, each followed by an ReLu activation function, a max-pooling layer, a dropout, and a batch normalization layer as a regularization technique. This is followed by 3 linear layers with a ReLu activation function and the final output layer of dimension 128x1 is considered as the final encoding of the input images.



```
EmbedNet(
  (cnn1): Sequential(
    (0): Conv2d(1, 64, kernel_size=(5, 5), stride=(1, 1))
    (1): ReLU(inplace)
    (2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (3): BatchNorm2d(2, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): Dropout2d(p=0.5)
    (5): Conv2d(4, 32, kernel_size=(5, 5), stride=(1, 1))
    (6): ReLU(inplace)
    (7): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (8): BatchNorm2d(2, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (9): Dropout2d(p=0.5)
    (10): Conv2d(8, 64, kernel_size=(5, 5), stride=(1, 1))
    (11): ReLU(inplace)
    (12): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (13): BatchNorm2d(2, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (14): Dropout2d(p=0.5)
    (15): Conv2d(16, 32, kernel_size=(5, 5), stride=(1, 1))
    (16): ReLU(inplace)
    (17): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (18): BatchNorm2d(2, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (19): Dropout2d(p=0.5)
  )
  (fc1): Sequential(
    (0): Linear(in_features=32768, out_features=500, bias=True)
    (1): ReLU(inplace)
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU(inplace)
    (4): Linear(in_features=500, out_features=132, bias=True)
  )
)
```

Fig. 12. Architecture of EmbedNet

IV. About the Datasets

The open source datasets provided by the University of Essex (face94, face95, face96) are used for the research work. Face94 has a total of 3060 images of 180x200 pixels of 153 individuals. Face95 has 1440 images of 72 individuals of 180x200 pixels and face 96 has 3040 images of 152 individuals of 196x196 pixels. These 24-bit RGB, JPEG images were combined to form a single dataset of 7540 images. Table I shows the composition of each dataset.

TABLE I. DATASET DESCRIPTION

Dataset Name	Total Images
Face 94	3060
Face 95	1440
Face 96	3016

V. Dataset Preparation for FDRENET

We used the above mentioned three datasets to test and train our model. Each dataset is divided into test and train sets of 20-80 images. Siamese dataset builder is used to creating around 1 lakh (0.1 million) Siamese pairs. Before loading the pairs, each image is subjected to HOG face detection technique and preprocessing to 64x64 grayscale. Figure 13 displays a subset of train images from the dataset. Siamese dataset builder creates an array of random true and false values with length equal to that of the dataset or simple the batch size. This array makes sure that we create a random dataset having both positive-positive and positive-negative image pairs. The rest of the steps for creating a suitable dataset for Siamese network training are briefly explained in Figure 14.

VI. Experiment

To train the Siamese network with EmbedNet as core model, we performed hyper-parameter tuning and decided to use Adam optimizer set to learning rate of 0.01. The Siamese pairs are fed to the network which is trained for 100 epochs. Graphs of iteration vs loss while training are presented in Figures 15-17. As Siamese network produces dissimilarity factor, we plotted a few test images with their dissimilarity factor printed on top. Figure 18 shows the results in terms of dissimilarity factor produced via trained FDREnet.

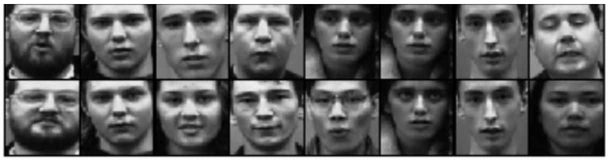
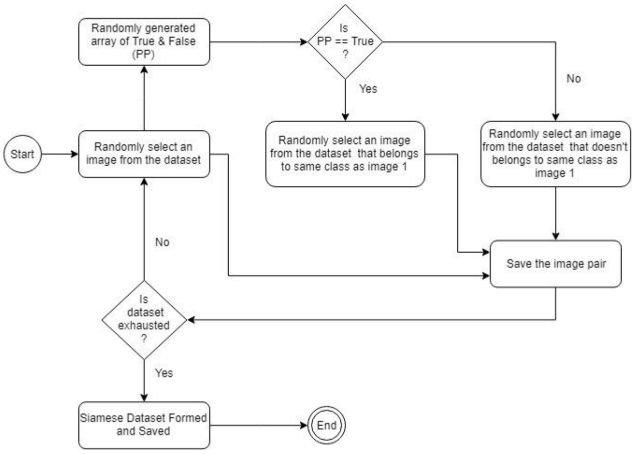


Fig. 13. Preprocessed dataset



Randomly generated array of True & False (PP)
Is PP == True ?
Yes
No
Start
Randomly select an image from the dataset
Randomly select an image from the dataset that belongs to same class as image 1
Randomly select an image from the dataset that doesn't belongs to same class as image 1
Save the image pair
No
Is dataset exhausted ?
Yes
Siamese Dataset Formed and Saved
End

Fig. 14. Siamese dataset builder

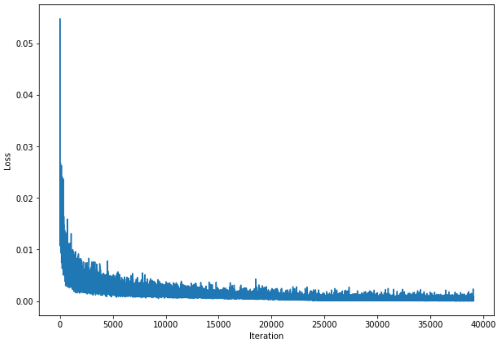


Fig. 15. Loss vs iteration graph while training face94

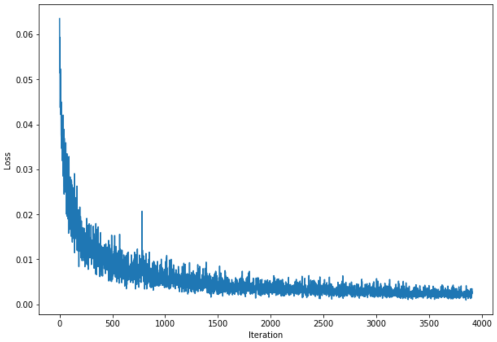


Fig. 16. Loss vs iteration graph while training face95

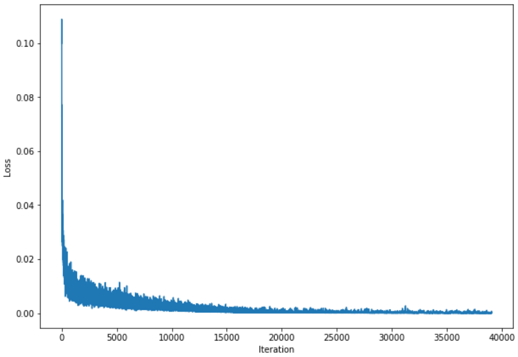


Fig. 17. Loss vs iteration graph while training face96

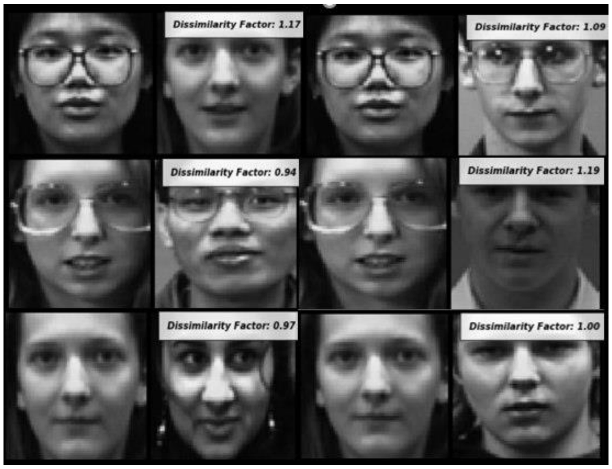

Dissimilarity Factor: 1.17
Dissimilarity Factor: 1.09
Dissimilarity Factor: 0.94
Dissimilarity Factor: 1.19
Dissimilarity Factor: 0.97
Dissimilarity Factor: 1.00

Fig. 18. Plotting dissimilarity factor for sample test images

VII. Results

Ideally, after training EmbedNet through FDREnet framework, it should be able to distinguish faces based on similarities and dissimilarities. We expect EmbedNet to produce results that are close for one person and far for different persons. If so, the embeddings predicted by EmbedNet for the training dataset can be clustered to find the accuracy of the classification. To determine the accuracy of this model we resort to clustering algorithms like k-nearest neighbor and support vector machine. For each case, the dataset is split into training set (80%) and test set (20%), and 25% of the training set is dedicated to validation. Tables II and III display the results of SVC and KNN clustering algorithms.

TABLE II. FDRENET ACCURACY BY SVM CLUSTERING

Data Set	Training Accuracy	Validation Accuracy	Test Accuracy
Face 94	99.68	99.36	98.03
Face 95	100.00	100.00	99.57
Face96	100.00	100.00	99.39

TABLE III. FDRENET BY KNN CLUSTERING

Data Set	Training Accuracy	Validation Accuracy	Test Accuracy
Face 94	99.89	98.88	97.83
Face 95	100.00	100.00	99.57
Face96	100.00	100.00	99.39

VIII. Conclusion

Using the proposed FDREnet, state of the art accuracy was achieved. Accuracy from clustering techniques justifies that the contrastive loss used with the deep learning model of the proposed FDREnet is successful in learning dissimilarities and similarities between faces in the given dataset, as well as classifying the faces into their labels (names).

Acknowledgement

We thank the University of Essex for the provision of face recognition data, namely for face94, face95 and face96 datasets (https://cswww.essex.ac.uk/mv/allfaces/) which are used for the research work in this paper.

References

[1] M. A. Rahman, I. K. E. Purnama, M. H. Purnomo, "Simple method of human skin detection using HSV and YCbCr color spaces", 2014 International Conference on Intelligent Autonomous Agents, Networks and Systems, Bandung, Indonesia, August 19-21, 2014

[2] J. Das, H. Roy, "Human Face Detection in Color Images Using HSV Color Histogram and WLD", 6th International Conference on Computational Intelligence and Communication Networks, Bhopal, India, November 14-16, 2014

[3] P. Viola, M. Jones, "Rapid object detection using a boosted cascade of simple features", 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, USA, December 8-14, 2001

[4] H. S. Dadi, G. K. M. Pillutla, "Improved Face Recognition Rate Using HOG Features and SVM Classifier", IOSR Journal of Electronics and Communication Engineering, Vol. 11, No. 4, pp. 34-44,2016

[5] O. Deniz, G. Bueno, J. Salido, F. D. L. Torre, "Face recognition using Histograms of Oriented Gradients", Pattern Recognition Letters, Vol. 32, No. 12, pp. 1598-1603, 2011

[6] U. Aiman, V. P. Vishwakarma, "Face recognition using modified deep learning neural network", 8th International Conference on Computing, Communication and Networking Technologies, Delhi, India, July 3-5, 2017

[7] W. Wang, J. Yang, J. Xiao, S. Li, D. Zhou, "Face Recognition Based on Deep Learning", in: Lecture Notes in Computer Science, Vol. 8944, pp. 812-820, Springer, 2014

[8] H. Kulkarni, G. Tofighi, "Unconstrained Facial Recognition using Supervised Deep Learning on Video", available at: https://www.researchgate.net/publication/325071878_Deep_Learning_for_Facial_Recognition, 2018

[9] F. Schroff, D. Kalenichenko, J. Philbin, "FaceNet: A unified embedding for face recognition and clustering", IEEE Conference on Computer Vision and Pattern Recognition, Boston, USA, June 7-12, 2015

[10] H. Wu, Z. Xu, J. Zhang, W. Yan, X. Ma, "Face recognition based on convolution siamese networks", 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, Shanghai, China, October 14-16, 2017

[11] N. H. Barnouti, W. E. Matti, S. S. M. Al-Dabbagh, M. A. S. Naser, "Face Detection and Recognition Using Viola-Jones with PCA-LDA and Square Euclidean Distance", International Journal of Advanced Computer Science and Applications, Vol. 7, No. 5, pp. 371-377, 2016

[12] P. Viola, M. Jones, "Rapid object detection using a boosted cascade of simple features", 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii, USA, December 8-14, 2001

[13] C. Shu. X. Ding, C. Fang, "Histogram of the Oriented Gradient for Face Recognition", Tsinghua Science and Technology, Vol. 16, No. 2, pp. 216-224, 2011

[14] H. S. Dadi, G. K. M. Pillutla, "Improved Face Recognition Rate Using HOG Features and SVM Classifier", IOSR Journal of Electronics and Communication Engineering, Vol. 11, No. 4, pp. 34-44, 2016